Scalability Analysis of a Satellite Signal Acquisition Algorithm inside the GNSS-SDR framework

Vinícius S. Fernandes*[1], Ana C. F. Vieira[1], Lucas F. Costa[1], Samuel X. de Souza[1]

[1]Departamento de Engenharia da Computação e Automação
Universidade Federal do Rio Grande do Norte (UFRN)
Natal – RN – Brazil

{vinicius.fernandes.058,ana.vieira.079,lucas.freire.705}@ufrn.edu.br

samuel@dca.ufrn.br

Abstract. *The Parallel Code Phase Search (PCPS) is a key algorithm in Global Navigation Satellite System (GNSS) receivers for detecting satellites and estimating signal parameters. This work analyzes its scalability in the GNSS-SDR framework, testing varying core counts (1–128) and problem sizes (2–32 million samples). Results show diminishing returns beyond 8–16 cores, with efficiency dropping below 10% at more than 32 cores, regardless of workload. The study identifies parallelization bottlenecks, suggesting optimizations are needed for real-time, high-channel GNSS Software-Defined Receiver (GNSS-SDR) applications on modern multi-core systems.*

1. Introduction

The acquisition step, among others in a Global Navigation Satellite System (GNSS) receiver, is known to be highly time-consuming [Presti 2009]. To solve this problem, many different algorithms have been developed, implementing different types of approaches to make this task faster. One of these algorithms is the Parallel Code Phase Search (PCPS) acquisition method. This method uses Fast Fourier Transforms (FFTs) to obtain the correlation results for all of the code phases simultaneously [J. Leclére; C. Botteron; Farine 2014]. One way to perform the acquisition of multiple satellite signals at a time is using channels. Each Channel encapsulates blocks for signal acquisition, tracking, and demodulation of the navigation message for a single satellite.

1.1. GNSS-SDR

The open-source Global Navigation Satellite System Software-Defined Receiver (GNSS-SDR) project is an initiative aimed at developing a run-time configurable receiver.

Parallel computing plays a vital role in GNSS-SDR, where real-time signal processing is required to acquire, track, and decode satellite signals. Key applications include:

• Signal acquisition: Parallel processing can significantly reduce the time required to detect and acquire satellite signals by distributing the computational load across multiple cores [Bui and Norris 2008].

• Correlation and tracking: Data-level parallelism can be applied to correlate incoming signals with locally generated replicas, enabling real-time tracking of multiple satellites [Quinn 2004].

1.2. Parallel Code Phase Search

The Parallel Code Phase Search is a fundamental acquisition algorithm in SDRs that enables rapid detection of satellite signals in the presence of noise and Doppler shifts. This approach leverages the FFT to efficiently correlate the received signal with locally generated replicas of the satellite code.

One of the key advantages of this technique is its ability to perform parallel searches over multiple code phase shifts simultaneously. This is achieved by exploiting the convolution theorem, which states that multiplication in the frequency domain corresponds to convolution in the time domain. As a result, the correlation peak corresponding to the correct code phase and Doppler frequency can be efficiently identified, enabling fast signal acquisition.

2. Methodology

The experiments were conducted on the Núcleo de Processamento de Alto Desempenho (NPAD) cluster using a single computational node equipped with two AMD EPYC 7713 processors running at 2.0 GHz, providing 128 physical cores. The cache hierarchy consists of 32 KB instruction and 32 KB data L1 cache per core, 512 KB L2 cache per core, and 256 MB shared L3 cache. The system is configured with 512 GB of DDR4 RDIMM memory distributed across 16 modules of 32 GB each.

The substantial cache resources (particularly the large 256 MB L3 cache) and high core count make this system particularly suitable for investigating parallel algorithm behavior at scale.

3. Results

Scalability refers to the ability of an algorithm to maintain or improve its efficiency as the number of processing units increases. By analyzing the scalability of PCPS, we aim to identify bottlenecks, optimize resource utilization, and provide insights into its applicability in real-time GNSS systems.

Therefore, the scalability of the PCPS algorithm inside the GNSS-SDR framework was analyzed by varying the number of processing units and measuring the corresponding speedup and efficiency, while using the default configuration file provided in the GNSS-SDR's documentation.

These tests were conducted by varying the number of CPU cores (from 1 to 128) and the number of acquisition channels (from 4 to 32), while fixating the sample size. Each configuration was executed 20 times, and the median was recorded.

The performance measurements are summarized in Table 1.

Channels	Number of Cores							
(acq_1c)	1	2	4	8	16	32	64	128
4	5.33	1.28	1.31	1.21	1.47	1.29	1.23	1.24
8	8.50	2.16	2.19	2.16	2.19	2.09	2.02	2.17
12	14.45	2.66	3.26	2.72	2.67	3.31	2.82	2.84
16	21.84	4.11	3.10	3.66	3.63	3.40	4.07	3.99
20	29.42	5.00	4.77	4.62	5.80	5.20	5.13	5.24
24	35.95	7.86	9.30	7.66	8.10	7.60	7.70	7.73
28	43.77	8.40	8.22	8.31	8.50	7.30	7.81	8.00
32	50.11	7.70	8.99	9.15	7.80	8.06	9.66	8.03

Table 1. Median execution time (s) of the Parallel Code Phase Search algorithm across different core and channel figurations and the default sample amount (2M).

To investigate further, efficiency measurements of the PCPS algorithm were conducted on the NPAD cluster described in Section 2 across different configurations of channel counts, fixed at 4 and 16 channels, core allocations, and sample sizes. The variation in sample sizes helps to grow the problem size and check for a scalable algorithm.

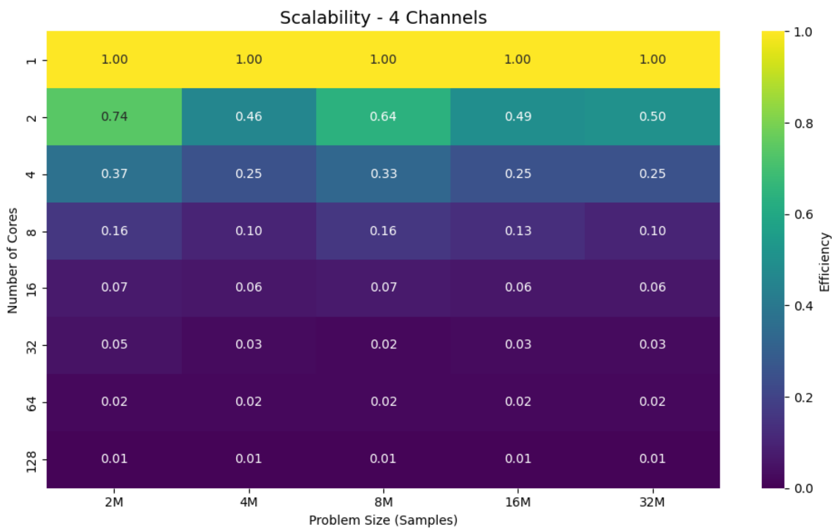

Figure 1. Efficiency for 4 channels across different core allocations and sample sizes.

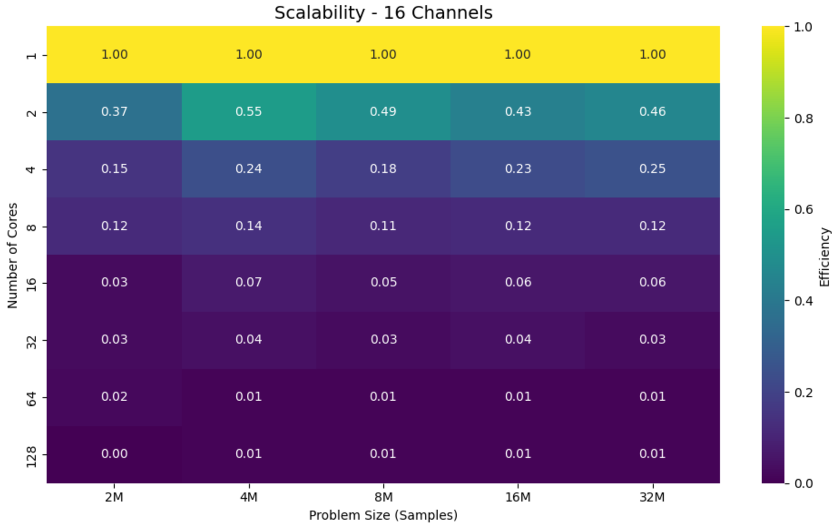

Figure 2. Efficiency for 16 channels across different core allocations and sample sizes.

It is noted that in most cases, the algorithm does not exhibit scalability beyond 8-16 cores, suggesting that parallel overhead becomes a limiting factor. The performance degradation is consistent and not due to outlier measurements. As demonstrated, PCPS fails to maintain efficiency across all tested problem sizes (2M to 32M samples) and channel configurations (4 to 16 channels).

4. Conclusion

Finally, the non-scalable architecture of PCPS, as demonstrated in Section 3, approximately maintains efficiency across all tested problem sizes (2M to 32M samples) with a fixed number of cores, but fails otherwise. This behavior is consistent across channel configurations (4 to 16 channels), with efficiency tables showing consistent degradation beyond 8 cores regardless of sample size. Optimal performance occurs at 4-8 cores, dropping to less than 10% at more than 8 cores, which limits hardware utilization and results in wasted computational resources when deployed on high-core-count nodes (more than 32 cores). Furthermore, this lack of scaling restricts its applicability in high-channel GNSS receivers with real-time constraints.

References

Bui, V. and Norris (2008). A component infrastructure for performance and power modeling of parallel scientific applications. In *CBHPC '08: Proceedings of the 2008 compFrame/HPC-GECO workshop on Component based high performance*, pages 1–11. ACM.

J. Leclére; C. Botteron; Farine, P.-A. (2014). Acquisition of modern gnss signals using a modified parallel code-phase search architecture. *Signal Processing*, 95:177–191.

Presti, L. L. e. a. (2009). Gnss signal acquisition in the presence of sign transition. *Journal of Selected Topics in Signal Processing*, 3:557–570.

Quinn, M. (2004). *Parallel Programming in C with MPI and OpenMP*. McGraw-Hill Higher Education, Boston.